10/531756

Rec'd PCT/PTO 14 APR 2005

ARRANGEMENT, SYSTEM AND METHOD FOR VECTOR PERMUTATION IN SINGLE-INSTRUCTION MULTIPLE-DATA MICROPROCESSORS

Field of the Invention

This invention relates to microprocessors with Single-Instruction Multiple-Data (SIMD) capability.

Background of the Invention

In the field of this invention microprocessors with SIMD architecture are arranged to process vector operands. It is known to provide instructions that permute (rearrange the order of) the components of vector operands in order to improve the efficiency of digital signal processing algorithms on SIMD microprocessors. Permutation parameters are required to determine the characteristics of the permutation to be performed.

However, this approach has the disadvantage(s) that if the vector permutation requires extra instructions, performance decreases. If the permutation parameters and/or the permuted vector operand require extra registers in the microprocessor's vector register file, a large register file is required. This increases the microprocessor's size and has a negative impact on program code density.

A need therefore exists for an arrangement, system and method for vector permutation in SIMD microprocessors wherein the abovementioned disadvantage(s) may be alleviated.

Statement of Invention

In accordance with a first aspect of the invention there is provided an arrangement for vector permutation in SIMD microprocessors as claimed in claim 1.

In accordance with a first aspect of the invention there is provided a system for vector permutation in SIMD microprocessors as claimed in claim 2.

In accordance with a third aspect of the invention there is provided a method for vector permutation in SIMD microprocessors as claimed in claim 5.

The arrangement preferably further includes a negate block coupled to the control means and coupled to receive and selectively negate vectors from the permutation logic block according to the control parameters received from the control means, wherein the control parameters include permutation parameters and negate parameters.

Preferably the control means includes at least one counter arranged to provide a sequential order for selecting one of the plurality of control registers.

The control register parameters are preferably also used for determining negate characteristics and the step of permutating further includes the step of selectively negating the vectors according to the parameters of the selected control register. Preferably the step of selecting further includes the following of a sequential order of the plurality of control registers.

Preferably the sequential order includes automatic sequencing through a set of fixed control parameters. Alternatively the sequential order preferably includes automatic sequencing through a set of programmable control parameters. The sequential order is preferably cyclical.

In this way an arrangement, system and method for vector permutation in SIMD microprocessors is provided in which no separate permutation instructions are necessary or need to be executed, and no permutation parameters need be stored in the vector registers. This leads to higher performance, a smaller vector register file and hence a smaller size of the microprocessor and better program code density.

Brief Description of the Drawings

One arrangement, system and method for vector permutation in SIMD microprocessors incorporating the present

invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

FIG. 1 shows a block schematic diagram of a known microprocessor with SIMD architecture; and

FIG. 2 shows a block schematic diagram of a microprocessor system with SIMD architecture incorporating the present invention.

Description of Preferred Embodiment(s)

Within the field of SIMD architecture, it is known that permutation and optional negate operations of vector operands may be performed as side operations of certain instructions and do not themselves require separate instructions or execution cycles.

However, programmers need control over when and how such permutations are performed. In order to control when permutations are performed, qualifiers are needed. These qualifiers may be:
- enable/disable mechanisms
- vector register numbers
- instruction types
- other

In order to control how permutations are performed, permutation parameters, source/destination operands or optional negate operations are needed. Such permutation

parameters can either be fixed (hard-wired for specific algorithms) or programmable (stored in registers).

Referring now to FIG. 1, there is shown a prior art microprocessor 5 with SIMD architecture. A vector register file 10 of the microprocessor feeds vector operands into a permutation logic block 20. The vector register file 10 has a predetermined number of registers. The number of general purpose and/or vector registers in modern Reduced Instruction Set Chip (RISC) machines typically is an integer to the power of 2 with 8/16/32/64 being the most common numbers. In the example depicted in FIG. 1, there are 32 128-bit registers, each register having four 32-bit elements. The last register (register 15) is used to store control parameters for controlling the permutation logic block 20, as depicted by arrow 17.

Referring now to FIG. 2, there is shown a microprocessor 100 with SIMD architecture. A read port of a vector register file 110 feeds vector operands into a permutation logic block 120 and from there into a negate logic block 130. The vector register file 110 has a predetermined number of registers. In the example depicted in FIG. 2, there are 8 128-bit registers (of which 5 are shown), each register having four 32-bit elements.

The output is typically used as source operand for a vector Arithmetic Logic Unit (ALU) (not shown).

Permutation and negate parameters relating to permutations to be performed upon the vectors of the vector register file 110 are stored as control parameters in a series of control registers 140. A control block 145 is coupled to each of the series of control registers 140 and is further coupled to provide the control parameters therefrom to control the permutation logic block 120 and the optional negate logic block 130. A counter 150 is also coupled to the control block 145, the counter 150 being arranged to determine which of the series of control registers is coupled via the control block 145 to the permutation logic block 120 and the optional negate logic block 130 at any one time.

In operation, the microprocessor 100 will commence with the counter 150 pointing at a given control register of the series 140, such as a first control register 141. When a permutation is to be performed (all qualifiers true), the control parameters (permutation and negate parameters) stored in the first control register 141 are provided via the control block 145 to the permutation logic block 120 and to the optional negate logic block 130. The contents of the vector register file 110 are then processed by the permutation logic block 120 and the optional negate logic block 130 according to these control parameters. It will be noted that the optional negate logic block 130, being optional, may or may not perform a negate function on the contents of the vector register file 110, depending upon the received control parameters.

Once processed, the output vector source operand is sent to the ALU (not shown) and the counter 150 is incremented. This causes the control block 145 to select the next control register of the series 140 (such as the second control register 142) for the next permutation. The counter 150 is arranged to cycle through each of the series of control registers 140 in a repeating manner.

It will be understood that the an arrangement, system and method for vector permutation in SIMD microprocessors described above provides the following advantages: No extra instructions are required to permute/negate the components of vector operands, leading to higher performance. Furthermore, no further registers of the vector register file are required to store the permuted/negated vector operands and the permutation parameters. It should be noted that even with programmable permutation parameters, the control registers 140 of FIG. 2 are significantly smaller than the vector register 15 of FIG. 1. Since the microprocessor's register file is smaller, this leads to a smaller size of the microprocessor and better program code density (fewer bits in op-codes for vector register addressing).

It will be appreciated by a person skilled in the art that alternative embodiments to that described above are possible. For example, the control register series 140 and counter 150 may be augmented by multiple counters and control register series, coupled with qualifiers such as instruction type or register number. Also the counting

sequence need not repeat in a cyclical fashion, and it is possible to load the counter(s) with specific sequence start points by adding just one further instruction. All of these features may be used to add complexity to the sequence of permutations and so further increase the flexibility of the architecture.

Furthermore the number and size of vector registers may differ from those described above, it being understood that the number of vector registers required by the present invention will be less than that required for an equivalent prior art arrangement.